

Adding a Constraint to EUROPA

A new project automatically includes its own module that is loaded into the EUROPA engine and stub C++ files to hold custom code (see [makeproject](#)). Therefore, extending EUROPA components simply requires C++ code be added to those existing files. Consider the Example project [here](#), which can be downloaded with:

```
svn co https://babelfish.arc.nasa.gov/svn/europa/benchmarks/trunk/Example Example
```

We have added an 'ExampleConstraint' that restricts a variable to have integer bounds. To create and use the constraint involved these steps:

1. Declare and define the constraint in [ExampleCustomCode.hh](#) and [ExampleCustomCode.cc](#).
2. Register the constraint. In [ModuleExample.cc](#):

```
CESchema* schema = (CESchema*)engine->getComponent("CESchema");  
  
REGISTER_CONSTRAINT(schema, ExampleConstraint, "example", "Default");
```

3. Use the constraint. In [Example-initial-state.nddl](#):

```
float x = [0.5, 3.5];  
example(x); // will be constrained to [1, 3];
```

4. Optionally, declare the constraint(this avoid warnings and enables parser type-checking). In [Example-model.nddl](#):

```
constraint example(x) { x <: numeric }
```

The result is that the global variable x will have range [1, 3] once EUROPA has been run. This range is output in the 'RUN_Example-planner_g_rt.Example-initial-state.xml.PlannerConfig?.xml.output' log file; notice these lines in [Example-Main.cc](#):

```
std::cout << "x lower bound: " << engine->getVariableByName("x")->getLowerBound() << std::endl;  
std::cout << "x upper bound: " << engine->getVariableByName("x")->getUpperBound() << std::endl;
```

For many other examples of how to write EUROPA constraints, see [Constraints.cc](#).